

**AMENDMENT TO THE CLAIMS:**

THIS LISTING OF CLAIMS WILL REPLACE ALL PRIOR VERSIONS, AND  
LISTINGS, OF CLAIMS IN THE APPLICATION.

**LISTING OF CLAIMS:**

1. (previously presented) A computer system employing management software written in a first computer language compatible with first software architecture and not compatible with second software architecture, said system comprising:
  - a schema formed within said first software architecture;
  - header files contained within said schema, said header files being represented in said first language and capable of being utilized by said management software;
  - means for manipulating said header files to locate public functions and/or data attributes of said header files;
  - means, responsive to operation of said manipulating means, for emitting code that calls said public functions and/or data attributes in said first language to obtain called public functions and/or data attributes; and,
  - means for converting said called public functions and/or data attributes to representations of said called public functions and/or data attributes formed in a different computer language compatible with said second software architecture.
2. (original) The computer system of claim 1 further comprising means for forwarding said representations to desired destinations within and beyond said system.

3. (original) The computer system of claim 1 and wherein said first computer language is RAID++ and said different computer language is XML/CIM.
4. (previously presented) The computer system of claim 1 and wherein said first computer language is an object-oriented language defining computer data and commands as objects, said manipulating means comprising:
  - means for opening at least one of said header files containing a declaration of at least one of said objects;
  - means for parsing said at least one of said header files to obtain name of class and name of parent class to which said at least one of said objects belongs; and,
  - means for creating a subroutine for accepting said at least one of said objects in said first computer language and generating the equivalent of said at least one of said objects in a different computer language compatible with said second software architecture.
5. (original) The computer system of claim 1 further comprising means for inhibiting initiation of operation of said converting means until said public functions and/or data attributes of said header files are located.
6. (original) The computer system of claim 1 further comprising means for initiating operation of said converting means upon locating the first of any one of said public functions and/or data attributes.

7. (original) The computer system of claim 1 and wherein said first computer language is C++ and said different computer language is XML/CIM.
8. (original) The computer system of claim 1 and wherein said first computer language is a first object-oriented language capable of pictorial representation typically in a parent-child tree configuration and said different computer language is a second object-oriented language capable of pictorial representation typically in a flat database configuration.
9. (original) The computer system of claim 1 further comprising means for inhibiting initiation of operation of said converting means until said public functions and/or data attributes of at least one of said header files are located.
10. (original) The computer system of claim 1 and wherein said management software is storage management software.
11. (previously presented) The computer system of claim 1 and wherein said management software is selected from the group consisting of storage, printer, and server management software.
12. (previously presented) A computer network employing a computer system utilizing management software written in a first computer language compatible with first

software architecture and not compatible with second software architecture, said network comprising:

a schema formed within said first software architecture;

header files contained within said schema, said header files being represented in said first language and capable of being utilized by said management software;

apparatus for manipulating said header files to locate public functions and/or data attributes of said header files; and,

apparatus, responsive to operation of said manipulating apparatus, for emitting code that calls said public functions and/or data attributes in said first language to obtain called public functions and/or data attributes and that converts said called public functions and/or data attributes to representations of said called public functions and/or data attributes formed in a different computer language compatible with said second software architecture.

13. (original) The network of claim 12 and wherein said computer system further comprises apparatus for forwarding said representations to desired destinations within and outside of said network.

14. (original) The network of claim 12 and wherein said first computer language is RAID++ and said different computer language is XML/CIM.

15. (previously presented) The network of claim 12 and wherein said first computer language is an object-oriented language defining computer data and commands as objects, said manipulating apparatus comprising:

apparatus for opening one of said header files containing a declaration of one of said objects;

apparatus for parsing said one of said header files to obtain name of class and name of parent class to which said one of said objects belongs; and,

apparatus for creating a subroutine for accepting said one of said objects in said first computer language and generating the equivalent of said one of said objects in a different computer language compatible with said second software architecture.

16. (original) The network of claim 12 further comprising apparatus for inhibiting initiation of operation of said converting apparatus until said public functions and/or data attributes of said header files are located.

17. (original) The network of claim 12 further comprising apparatus for initiating operation of said converting apparatus upon locating the first of any one of said public functions and/or data attributes.

18. (original) The network of claim 12 and wherein said first computer language is C++ and said different computer language is XML/CIM.

19. (original) The network of claim 12 and wherein said first computer language is a first object-oriented language capable of pictorial representation typically in a parent-child tree configuration and said different computer language is a second object-oriented language capable of pictorial representation typically in a flat database configuration.

20. (original) The network of claim 12 further comprising apparatus for inhibiting initiation of operation of said converting apparatus until said public functions and/or and data attributes of at least one of said header files are located.

21. (original) The network of claim 12 further comprising a SAN which communicates with and is controlled by said computer system.

22. (original) The network of claim 12 and wherein said management software is storage management software.

23. (previously presented) The network of claim 12 and wherein said management software is selected from the group consisting of storage, printer, and server management software.

24. (previously presented) A method for utilizing standardized software architecture to be practiced in a computer system employing management software written in a first

computer language compatible with first software architecture and not compatible with said standardized software architecture, said method comprising:

    said management software utilizing a schema having header files in said first language;

    manipulating said header files to locate public functions and/or data attributes of said header files; and,

    responsive to operation of said manipulating, emitting code that calls said public functions and/or data attributes in said first language to obtain called public functions and/or data attributes and converts said called public functions and/or data attributes to representations of said called public functions and/or data attributes formed in a different computer language compatible with said standardized software architecture.

25. (original) The method of claim 24 further comprising forwarding said representations to desired destinations within and beyond said system.

26. (original) The method of claim 25 and wherein said first computer language is RAID++ and said different computer language is XML/CIM.

27. (previously presented) The method of claim 25 and wherein said first computer language is an object-oriented language defining computer data and commands as objects, said manipulating comprising:

    opening one of said header files containing a declaration of one of said objects;

parsing said one of said header files to obtain name of class and name of parent class to which said one of said objects belongs; and,

creating a subroutine for accepting said one of said objects in said first computer language and generating the equivalent of said one of said objects in a different computer language compatible with said standardized software architecture.

28. (original) The method of claim 27 further comprising inhibiting initiation of operation of said converting until said public functions and/or data attributes of said header files are located.

29. (original) The method of claim 28 further comprising initiating operation of said converting upon locating the first of any one of said public functions and/or data attributes.

30. (original) The method of claim 29 and wherein said first computer language is C++ and said different computer language is XML/CIM.

31. (original) The method of claim 27 and wherein said first computer language is a first object-oriented language capable of pictorial representation typically in a parent-child tree configuration and said different computer language is a second object-oriented language capable of pictorial representation typically in a flat database configuration.

32. (original) The method of claim 24 further comprising inhibiting initiation of operation of said converting until said all public function and data attributes of at least one of said header files are located.

33. (original) The method of claim 31 further comprising inhibiting initiation of operation of said converting until said public functions and/or and data attributes of at least one of said header files are located.

34. (previously presented) The method of claim 24 and wherein said standardized software architecture is preferred non-legacy software architecture.

35. (original) The method of claim 24 and wherein said management software is storage management software.

36. (previously presented) The method of claim 24 and wherein said management software is selected from the group consisting of storage, printer, and server management software.

37. (previously presented) A computer program product including management software written in a first language and embodied in a computer system for operation on said computer system designed in accordance with first software architecture and not

compatible with other than said first software architecture, said computer program product comprising:

programmable code for utilizing a schema having header files in said first language;

programmable code for manipulating said header files to locate public functions and/or data attributes of said header files; and,

programmable code, responsive to said manipulating, for emitting special code that calls said public functions and/or data attributes in said first language to obtain called public functions and/or data attributes and converts said called public functions and/or data attributes to representations of said called public functions and/or data attributes formed in a different computer language compatible with said other than said first software architecture.

38. (original) The computer program product of claim 37 further comprising programmable code for forwarding said representations to desired destinations within and beyond said system.

39. (previously presented) The computer program product of claim 38 and wherein said first computer language is an object-oriented language defining computer data and commands as objects, said programmable code for manipulating comprising:

programmable code for opening one of said header files containing a declaration of one of said objects;

programmable code for parsing said one of said header files to obtain name of class and name of parent class to which said one of said objects belongs; and,

programmable code for creating a subroutine for accepting said one of said objects in said first computer language and generating the equivalent of said one of said objects in a different computer language compatible with said other than said first software architecture.

40. (original) The computer program product of claim 39 and wherein said first computer language is RAID++ and said different computer language is XML/CIM.

41. (original) The computer program product of claim 40 further comprising programmable code for inhibiting initiation of operation of said programmable code for converting until said public functions and/or data attributes of said header files are located.

42. (original) The computer program product of claim 40 further comprising programmable code for initiating operation of said programmable code for converting upon locating the first of any one of said public functions and/or data attributes.

43. (original) The computer program product of claim 39 and wherein said first computer language is C++ and said different computer language is XML/CIM.

44. (original) The computer program product of claim 37 and wherein said first computer language is a first object-oriented language capable of pictorial representation typically in a parent-child tree configuration and said different computer language is a second object-oriented language capable of pictorial representation typically in a flat database configuration.

45. (previously presented) The computer program product of claim 44 further comprising programmable code for inhibiting initiation of operation of said programmable code for converting until said public functions and/or and data attributes of at least one of said header files are located.

46. (previously presented) The computer program product of claim 37 and wherein said first software architecture is legacy software architecture and said other than said first software architecture is preferred non-legacy software architecture.

47. (original) The computer program product of claim 37 and wherein said management software is storage management software.

48. (previously presented) The computer program product of claim 37 and wherein said management software is selected from the group consisting of storage, printer, and server management software.

49. (previously presented) A computer program product compatible with preferred non-legacy software architectures and operating in a computer system employing management software written in a first computer language compatible with legacy software architecture and not compatible with said preferred non-legacy software architectures, said computer program product embodied in said system and comprising:

programmable code for utilizing a schema having header files in said first language;

programmable code for manipulating said header files to locate public functions and/or data attributes of said header files;

programmable code, responsive to said manipulating, for emitting special code that calls said public functions and/or data attributes in said first language to obtain called public functions and/or data attributes; and,

programmable code for converting said called public functions and/or data attributes to representations of said called public functions and/or data attributes formed in a plurality of different computer languages each being compatible with at least one of said preferred non-legacy software architectures.

50. (original) The computer program product of claim 49 and wherein said management software is storage management software.

51. (previously presented) The computer program product of claim 49 and wherein said management software is selected from the group consisting of storage, printer, and server management software.

52. (previously presented) In a computer network including a computer system having a functional system therein with management software including a schema for managing said functional system under control of said computer system in accordance with first software architecture, a translator-compiler embodied in said computer system for permitting communication about said managing said functional system to be transmitted between said computer system and computer devices operating under second software architecture, said translator-compiler comprising:

program code for accessing header files within said schema to obtain a header file containing particular information;

program code for parsing said header file to obtain a particular result;

program code for opening an output file for storage of other than said particular information related to said particular result;

program code for continued parsing of said header file to locate public functions and/or data attributes; and,

program code for emitting special code to said output file that calls said public functions and/or data attributes to obtain called public functions and/or data attributes and for converting said called public functions and/or data attributes to language compatible with said second software architecture;

whereby communication about managing said functional system is transmitted between said computer system and said computer devices operating under said second software architecture.

53. (original) In the computer network of claim 52 and wherein said functional system is a storage system and said management software is storage management software.
54. (original) In the computer network of claim 52 and wherein said functional system is a SAN and said management software is SAN management software.
55. (previously presented) In the computer network of claim 52 and wherein said first software architecture is legacy software architecture and said second software architecture is non-legacy software architecture.
56. (original) In the computer network of claim 52 and wherein said functional system is selected from the group consisting of storage system, printer system, server system or other-component system and said management software is selected from the group consisting of storage management software, printer management software, server management software and other-component management software respectively.
57. (previously presented) In the computer network of claim 52 and wherein said computer devices are located within said network.
58. (previously presented) In the computer network of claim 52 and wherein said computer devices are located outside of said network.

59. (previously presented) A method to be practiced on a computer system in a computer network including a functional system controlled by said computer system compatible with legacy software architecture having header files, said method comprising:

receiving and manipulating said header files;  
receiving first requests in first language incompatible with said legacy software architecture;  
obtaining responses to said first requests in second language compatible with said legacy software architecture; and,  
converting said responses to equivalent responses compatible with said first language; and  
communicating said equivalent responses to the destination from which, or to destinations related to that from which, said first requests originated.

60. (previously presented) The method of claim 59 and wherein said functional system is a storage system.

61. (previously presented) The method of claim 59 and wherein said functional system is a SAN.

62. (previously presented) The method of claim 59 and wherein said first requests are received from outside of said network.

63. (previously presented) In a computer system compatible with computer software architecture, management software embodied in said computer system for controlling at least one processor in said system to perform a method of responding to requests, said method comprising:

receiving first requests in first language incompatible with said computer software architecture;

obtaining responses to said first requests in second language compatible with said computer software architecture; and,

converting said responses to equivalent responses compatible with said first language and communicating said equivalent responses to the destination from which, or to destinations related to that from which, said first requests originated.

64. (previously presented) In the computer system of claim 63 and wherein said computer software architecture is legacy software architecture.

65. (original) In the computer system of claim 64 and wherein said management software is storage management software.

66. (original) In the computer system of claim 64 and wherein said management software is SAN management software.

67. (original) In the computer system of claim 66 and wherein said destination is located outside of said computer system.

68. (previously presented) A computer program product embodied in a computer compatible with computer software architecture comprising:

programmable code for receiving first requests in first language incompatible with said computer software architecture;

programmable code for obtaining responses to said first requests in second language compatible with said computer software architecture; and,

programmable code for converting said responses to equivalent responses compatible with said first language and for communicating said equivalent responses to the destination from which, or to destinations related to that from which, said first requests originated.

69. (previously presented) The computer program product of claim 68 and wherein said computer software architecture is legacy software architecture.

70. (previously presented) A method for managing functional systems to be practiced on a computer compatible with computer software architecture comprising:

receiving first requests in first language incompatible with said computer software architecture;

obtaining responses to said first requests in second language compatible with said computer software architecture; and,

converting said responses to equivalent responses compatible with said first language and communicating said equivalent responses to the destination from which, or to destinations related to that from which, said first requests originated.

71. (previously presented) The method of claim 70 and wherein said computer software architecture is legacy software architecture.

72. (original) The method of claim 71 and wherein said functional systems include a storage system.

73. (original) The method of claim 71 and wherein said functional systems include a SAN.

74. (previously presented) The method of claim 71 and wherein said functional systems are selected from the group consisting of storage systems, printer systems, and server systems.

75. (previously presented) In a computer network including a computer system and a storage system controlled by said computer system, a method for managing storage compatible with software architecture having header files, said method being deployed on both said computer system and said storage system, said method comprising:

translating and manipulating said header files to obtain translated and manipulated header files;

receiving first requests from outside of said network in first language incompatible with said software architecture;

in cooperation with said translated and manipulated header files, obtaining responses to said first requests in second language compatible with said software architecture; and,

in cooperation with said translated and manipulated header files, converting said responses to equivalent responses compatible with said first language and communicating said equivalent responses to said outside of said network.

76. (previously presented) The method of claim 75 and wherein said software architecture is legacy software architecture.

77. (original) In the computer network of claim 76 further comprising said storage system is a SAN.

78. (original) In the computer network of claim 75 and wherein said first language is a first object-oriented language capable of pictorial representation typically in a flat database configuration and said second language is a second object-oriented language capable of pictorial representation typically in a parent-child tree configuration.

79. (original) In the computer network of claim 78 and wherein said first language is CIM/XML and said second language is C++.

80. (original) In the computer network of claim 79 and wherein said C++ language is RAID++.

81. (currently amended) In an improved network including a first computer network operating in accordance with first software architecture and a second computer network operating in accordance with second software architecture, the improvement comprising:

an interface between said first computer network and said second computer network to automatically convert communication from said second computer network into a form compatible with said first computer network, and to automatically convert response to said communication generated by said first computer network into a form compatible with said second computer network;

wherein said first software architecture is legacy software architecture and said second software architecture is non-legacy software architecture.

82. (canceled)

83. (currently amended) The improvement of claim [[82]]81 and wherein said first computer network operates in accordance with said legacy software architecture supporting a first object-oriented computer language capable of pictorial representation typically in a parent-child tree configuration, and wherein said second computer network operates in accordance with said non-legacy software architecture supporting a second object-oriented computer language capable of pictorial representation typically in a flat database configuration.

84. (original) The improvement of claim 83 and wherein said first object-oriented computer language is C++ and wherein said second object-oriented computer language is XML/CIM.

85. (original) The improvement of claim 84 and wherein said communication includes management software communication.

86. (original) The improvement of claim 85 and wherein said management software communication includes storage management software communication.

87. (original) The improvement of claim 86 and wherein said storage management software communication relates to SAN communication.

88. (previously presented) The improvement of claim 85 and wherein said management software communication includes storage, printer, and server communications.

89. (original) The improvement of claim 81 and wherein said response is communicated to the destination from which, or to destinations related to that from which, said communication originated.